MODELOS DE PROCESO DE CICLO DE VIDA DE SOFTWARE (PM-SDLC): EVOLUCIÓN Y ELEMENTOS COMUNES.

Laura Cecilia Rodríguez Martínez[1]
Dr. Manuel Mora T.[1]
Dr. Francisco Álvarez R.[1]
Dr. Miguel Vargas M.[1]

INTRODUCCIÓN

Los modelos de proceso de ciclo de vida de desarrollo de software (**PM-SDLC**) guían el desarrollo de las aplicaciones de software, estas aplicaciones son productos complejos y difíciles de desarrollar y evaluar, por ello, uno de los principales objetivos de la investigación sobre PM-SDLC es el estudio y mejoramiento de dichos procesos. Esta investigación trata de identificar los componentes de los PM-SDLC y clarificar sus interrelaciones.

Un **PM-SDLC** representa una secuencia encadenada de actividades, objetos, transformaciones, y eventos que componen estrategias para completar la evolución del software [4]. Este modelo abarca lo siguiente: (i) **Especificación**; (ii) **Análisis y Diseño**; (iii) **Codificación**; (iv) **Prueba e implementación**. Son base de esta investigación, los modelos más representativos propuestos a través de la historia del desarrollo de software en la literatura científica [5] (Waterfall, SADT, Prototyping, Structured Cycle, Spiral, RAD, Win-Win Spiral Unified Process, PSP, TSP, MBASE, XP, CBASE). A través del tiempo se han propuesto una serie de modelos de procesos de ciclo de vida de desarrollo de sistemas intensivos en software, que en general son una evolución unos a partir de otros. Los conceptos base de este trabajo son: (i) **Ingeniería de Software.** Es la aplicación de un planteamiento sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software [1]. (ii) **Procesos software.** Los elementos del Proceso de Software (PS) son fases, actividades, productos o artefactos, roles y agentes [2]. (iii) **Sistemas intensivos en Software**. Se refiere a la emergencia de todo tipo de sistemas formados por componentes de software, hardware, información, personas, procedimientos y su entorno mismo, donde el componente de software es crítico. [3]

OBJETIVOS

Cada PM-SDLC propone una serie de componentes específicos, que son distintos en cada uno de ellos, esta investigación permitirá identificar dichos componentes y clarificar sus interrelaciones, mostrando los elementos comunes entre los PM-SDLC que tienen su origen en las distintas disciplinas base de esta investigación (Ingeniería de Software, Ingeniería de Sistemas y Sistemas de Información).

MATERIALES Y MÉTODOS

Los materiales a utilizar son: (i) Literatura actual científica sobre Ciclos de Vida de Sistemas de Software; (ii) Literatura con la propuesta original de cada modelo de Ciclos de Software. Los métodos a utilizar son: (i) Revisión y Análisis de Literatura; (ii) Clasificación de la Información e (iii) Inferencia Lógica.

RESULTADOS Y CONCLUSIONES

La principal ventaja de este trabajo contra otros es la comparación sistemática en base a una meta-modelo de **proceso de software** genérico [2]. Se construyó el marco (*framework*) de comparación basado en este meta-modelo, y se aplicó identificando los elementos comunes entre los distintos PM-SDLC:

- Todos los PM-SDLC definen actividades encaminadas a conocer las necesidades de los usuarios involucrados (*stakeholders*). En los primeros modelos estas actividades se limitaba al levantamiento y especificación de requerimientos; y posteriormente se llega hasta el modelado de negocios y construcción de prototipos.
- Todos los PM-SDLC definen actividades encaminadas a definir el alcance del sistema. En los primeros modelos estas actividades se limitaban a la determinación de objetivos, restricciones y alternativas; y en los modelos más recientes se llega hasta el análisis de riesgos y condiciones "win-win".
- Todos los PM-SDLC incluyen actividades encaminadas a definir la arquitectura del sistema (definición de requerimientos de sistema, diseño de arquitectura). En los modelos iniciales estas actividades se limitaba a la definición de requerimientos de sistema.
- Actividades de construcción del software: análisis, diseño, codificación, prueba, implementación.
- Los modelos que manejan un prototipo previo al sistema, no tienen un producto terminado intermedio, dado que manejan el prototipo: *Waterfall, SADT, Prototyping, RAD.*
- Los tres modelos más completos incluyen de manera explícita la elaboración de un manual de usuario: *Waterfall, Structured Analysis, Unified Process, MBASE*.

En respuesta al avance tecnológico y tratando de reforzar lo que se considere una debilidad en un PM-SDLC específico, cada nuevo PM-SDLC que surge utiliza como base los PM-SDLC existentes. El surgimiento de nuevos modelos evolucionados a partir de otros anteriores, indica el avance científico y la necesidad de la aplicación de procesos para el desarrollo de sistemas de software exitosos.

BIBLIOGRAFÍA

[1] IEEE (1990). IEEE Std 610.12-1990, Standard Glossary of Software Engineering.

[2] Oktaba, H.,.Ibargüengoitia, G. (1998). Software process Modeled with Objects: Static View, Computación y Sistemas Vol. 1 No. 4 pp. 228-238.

[3] Mora, M. et al. (2006). The Interaction of Systems Engineering, Software Engineering and Information Systems disciplines: A System-Oriented View. Accepted for the 18th. International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, August 7-12.

[4] Scacchi, W. (2001). Process Models in Software Engineering. In: J.J. Marciniak (ed.), Encyclopedia of Software Engineering, Wiley, New York

[5] Sommerville, I. (1995). Ingeniería de Software, 5a. Ed., Capítulo 1. McGraw-Hill.

[1] Universidad Autónoma de Aguascalientes